Homework 3

Name

4.1 Comparison of proportions

A randomized experiment is performed within a survey. 1000 people are contacted. Half the people contacted are promised a 5 dollar incentive to participate, and half are not promised an incentive. The result is a 50% response rate among the treated group and 40% response rate among the control group. Give an estimate and standard error of the average treatment effect.

```
N = 1000
n1 = 1000/2
n2= 1000/2
resp1 = 0.5
resp2= 0.4

mean = resp1 - resp2

se = sqrt(0.5^2/500 + 0.5^2/500)

paste("Mean: ", mean, ", Standard Error: ", round(se,2), sep="")
```

```
## [1] "Mean: 0.1, Standard Error: 0.03"
```

4.2 Choosing sample size

You are designing a survey to estimate the gender gap: the difference in support for a candidate among men and women. Assuming the respondents are a simple random sample of the voting population, how many people do you need to poll so that the standard error is less than 5 percentage points?

We're looking for candidatesupport, which would be present or not present. This can be approximated with a binomial distribution, which has a variance of $p(1-p)$. Coincidentally, this value also maximizes when $p = 0.5$, or, substituting, $\sigma^2 = 0.5 * (1 - 0.5) = 0.25$. If we want to make sure our standard of error will be less than 5%, we can use this to consider our case with the highest possible standard error.

The equation for standard error is $SE = \frac{\sigma}{\sqrt{n}}$. Now, we want to solve for n:

$SE = \frac{\sigma}{\sqrt{n}}, \sqrt{n} = \frac{\sigma}{SE}, n = (\frac{\sigma}{SE})^2 = \frac{\sigma^2}{SE^2}$

Let's calculatE:

```
sigma2 = 0.5*(1-0.5)
SE = 0.05

n = sigma2/SE^2
paste("We want to poll",n,"people.")
```

```
## [1] "We want to poll 100 people."
```

4.4 Designing an experiment

You want to gather data to determine which of two students is a better basketball shooter. You plan to have each student take N shots and then compare their shooting percentages. Roughly how large does N have to be for you to have a good chance of distinguishing a 30% shooter from a 40% shooter?

The standard error of difference in two proportion is $sqrt(p_1(1-p_1)+p_2(1-p_2))$. Let us interpret good chance as 80%. We can use sample size calculation from P.296, which results in calculation:

$$n = (p_1(1-p_1)+p_2(1-p_2))(2.8/(p_1-p_2))^2$$

```
(0.3*0.7+0.4*0.6)*(2.8/(0.3-0.4))^2
```

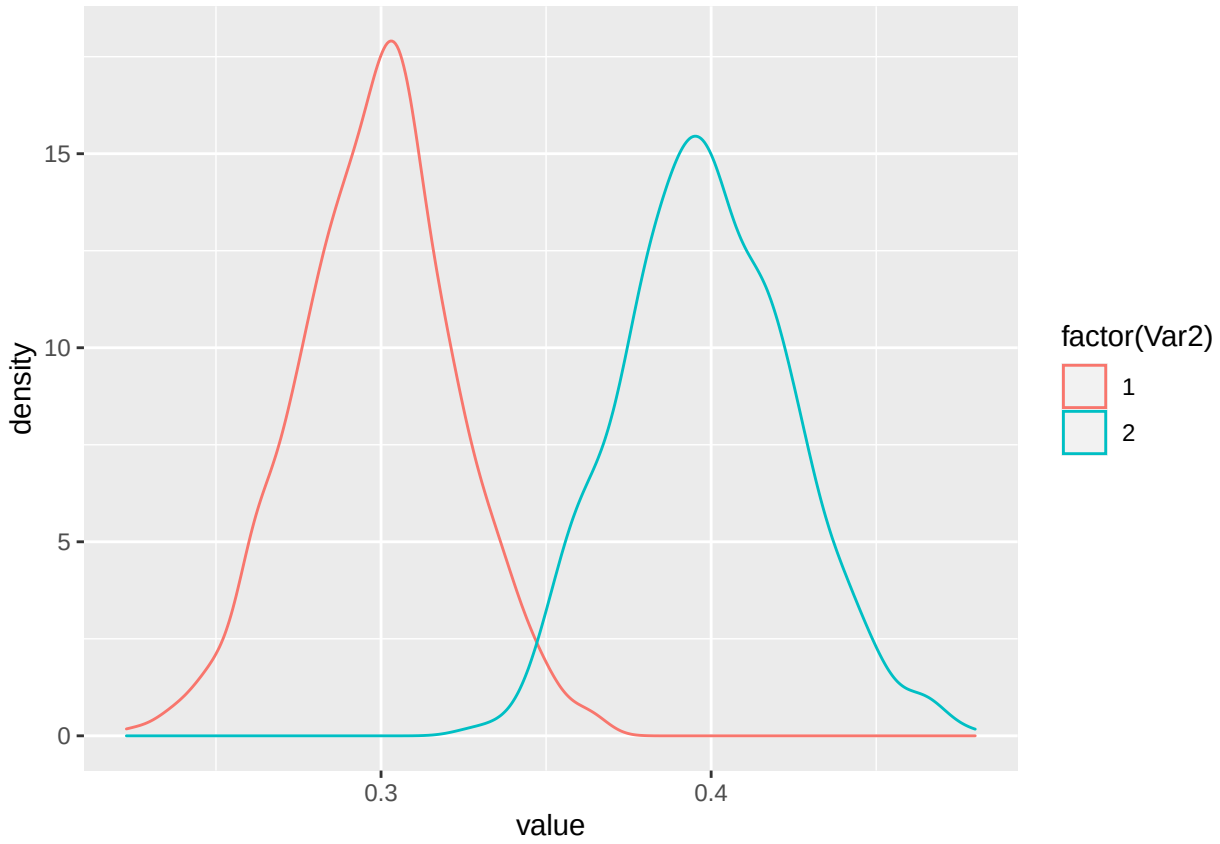
```
## [1] 352.8
```

So the result is about 350 shots.

```
sample_size = 350
resmat<-matrix(NA,2,1000)
sigvec<-rep(NA,1000)
for(i in 1:1000){
  ashot<-rbinom(sample_size,1,0.3)
  bshot<-rbinom(sample_size,1,0.4)
  resmat[1,i]<-p2<-mean(ashot)
  resmat[2,i]<-p1<-mean(bshot)
  sigvec[i]<-abs(p1-p2)-abs(qnorm(0.025,0,1))*sqrt(p1*(1-p1)/sample_size+p2*(1-p2)/sample_size) > 0
}
mean(sigvec)
```

```
## [1] 0.803
```

```
ggplot(melt(t(resmat)))+geom_density()+aes(x=value,color=factor(Var2))
```



4.6 Hypothesis testing

The following are the proportions of girl births in Vienna for each month in Girl births 1908 and 1909 (out of an average of 3900 births per month):

```
birthdata <- c(.4777,.4875,.4859,.4754,.4874,.4864,.4813,.4787,.4895,.4797,.4876,.4859,
               .4857,.4907,.5010,.4903,.4860,.4911,.4871,.4725,.4822,.4870,.4823,.4973)
```

The data are in the folder Girls. These proportions were used by von Mises (1957) to support a claim that that the sex ratios were less variable than would be expected under the binomial distribution. We think von Mises was mistaken in that he did not account for the possibility that this discrepancy could arise just by chance.

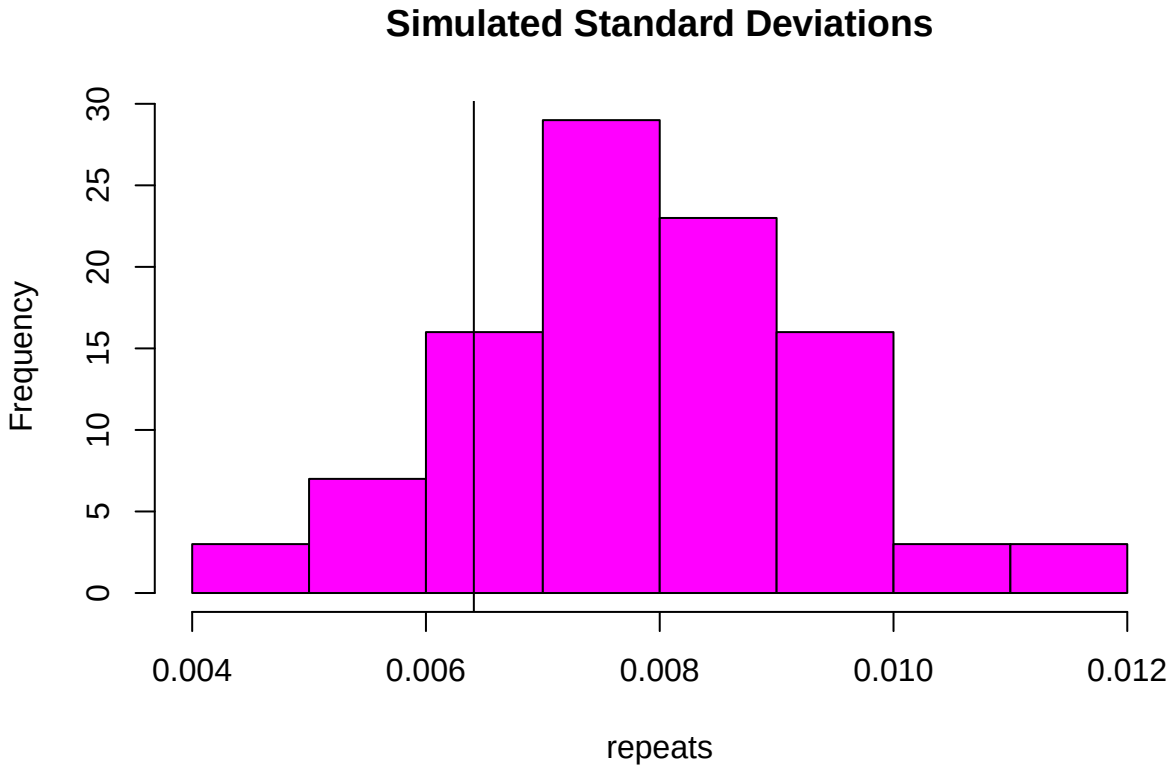
(a) Compute the standard deviation of these proportions and compare to the standard deviation that would be expected if the sexes of babies were independently decided with a constant probability over the 24-month period.

```
samplesd = sd(birthdata)
constantsd = sd(rbinom(20,3900,0.5)/3900)

paste("Data Standard Deviation:",round(samplesd,5),
      "Constant Standard Deviation:",round(constantsd,5))
```

```
## [1] "Data Standard Deviation: 0.00641 Constant Standard Deviation: 0.00677"
```

(b) The observed standard deviation of the 24 proportions will not be identical to its theoretical expectation. In this case, is this difference small enough to be explained by random variation? Under the randomness model, the actual variance should have a distribution with expected value equal to the theoretical variance, and proportional to a chi-square random variable with 23 degrees of freedom; see page 53.

```
repeats <- 1:100

for(i in 1:100)
  repeats[i] = sd(rbinom(20,3900,mean(birthdata))/3900)

hist(repeats,col="magenta",main = "Simulated Standard Deviations")
abline(v=samplesd)
```



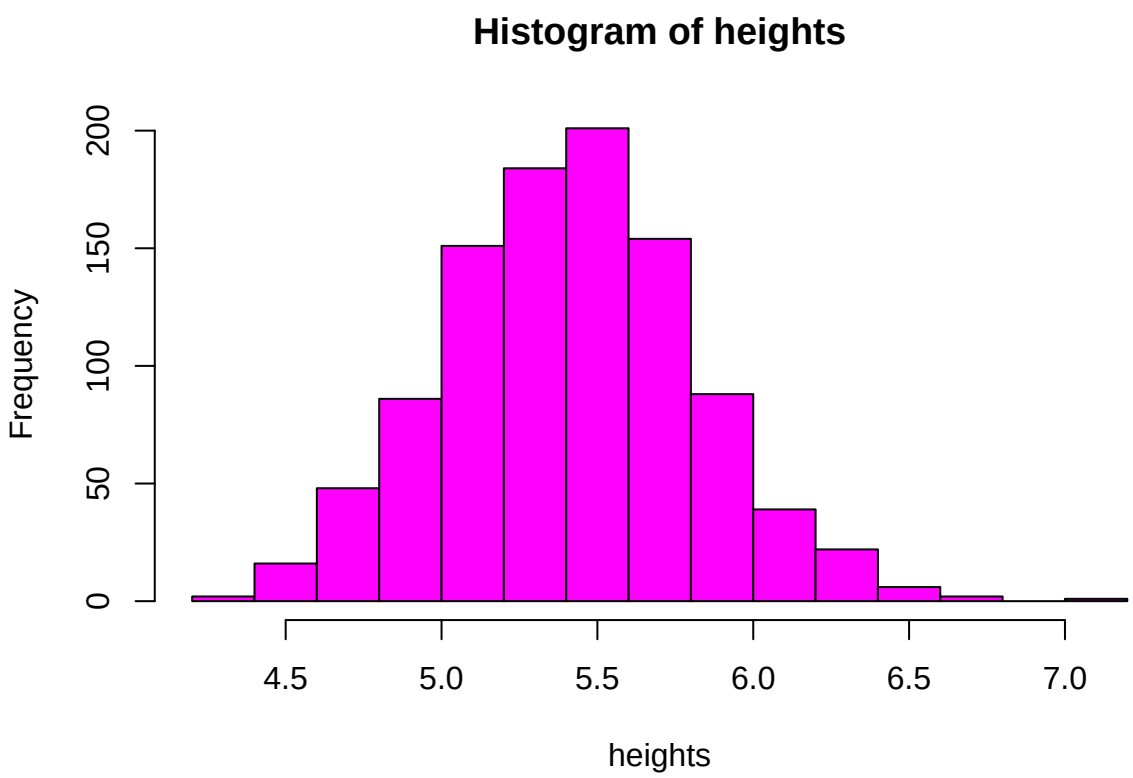
Simulating from a random sample, the data's standard deviation doesn't seem to be too far from the normal distribution's, and implies it can be up to chance.

5.5 Distribution of averages and differences

The heights of men in the United States are approximately normally distributed with mean 69.1 inches and standard deviation 2.9 inches. The heights of women are approximately normally distributed with mean 63.7 inches and standard deviation 2.7 inches. Let x be the average height of 100 randomly sampled men, and y be the average height of 100 randomly sampled women. In R, create 1000 simulations of x - y and plot their histogram. Using the simulations, compute the mean and standard deviation of the distribution of x - y and compare to their exact values.

```
heights = 1:1000

for ( i in 1:1000){
```

```
  x <- mean(rnorm(100,69.1,2.9))
  y <- mean(rnorm(100,63.7,2.7))
  heights[i] = x - y
}

hist(heights,col="magenta")
```




```
paste("Simulated Mean Difference:",round(mean(heights,3)),", Simulated Standard Deviation:", round(sd(h
```

```
## [1] "Simulated Mean Difference: 5 , Simulated Standard Deviation: 0.4"
```

```
paste("True Mean:",round(69.1-63.7,2),"True Standard Deviations: ", round(sqrt(2.9^2/100+2.7^2/100),2))
```

```
## [1] "True Mean: 5.4 True Standard Deviations:  0.4"
```

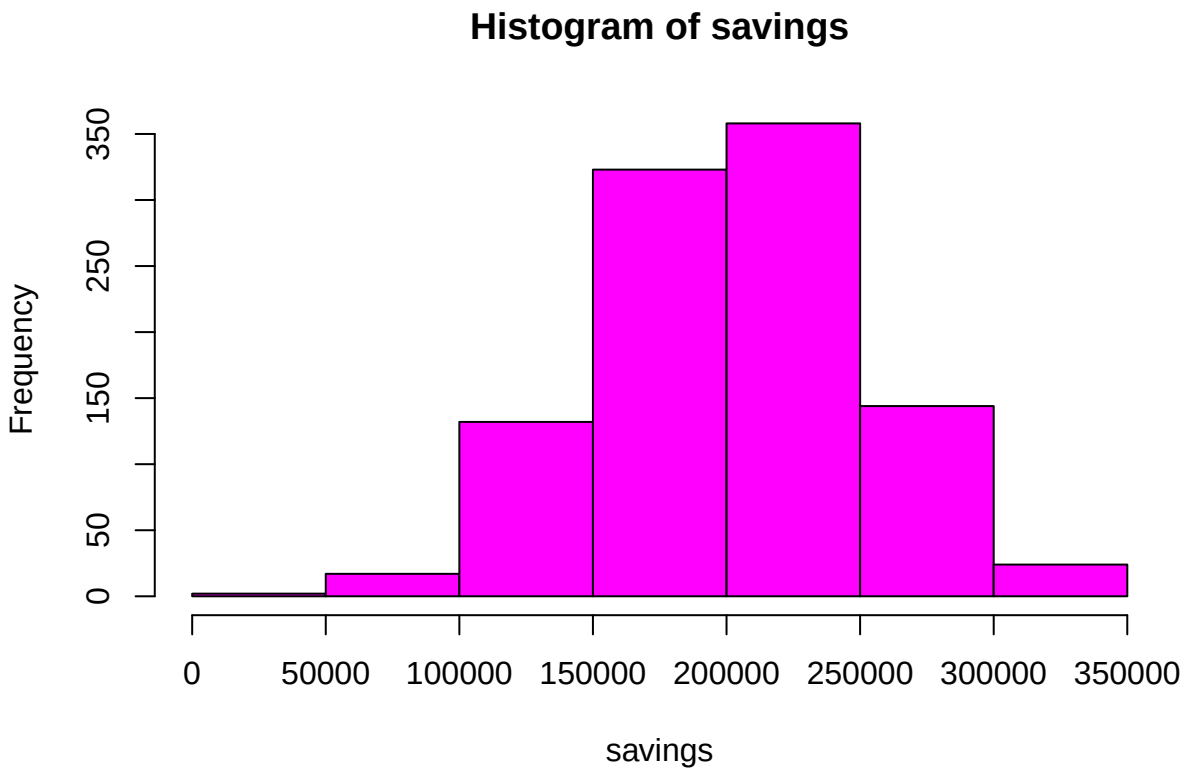
5.6 Propagation of uncertainty:

We use a highly idealized setting to illustrate the use of simulations in combining uncertainties. Suppose a company changes its technology for widget production, and a study estimates the cost savings at 5 dollars per unit, but with a standard error of 4 dollars. Furthermore, a forecast estimates the size of the market (that is, the number of widgets that will be sold) at 40 000, with a standard error of 10 000. Assuming these two sources of uncertainty are independent, use simulation to estimate the total amount of money saved by the new product (that is, savings per unit, multiplied by size of the market).

This study says the product can save you negative money 10.56 percentage of the time and that bothers me.

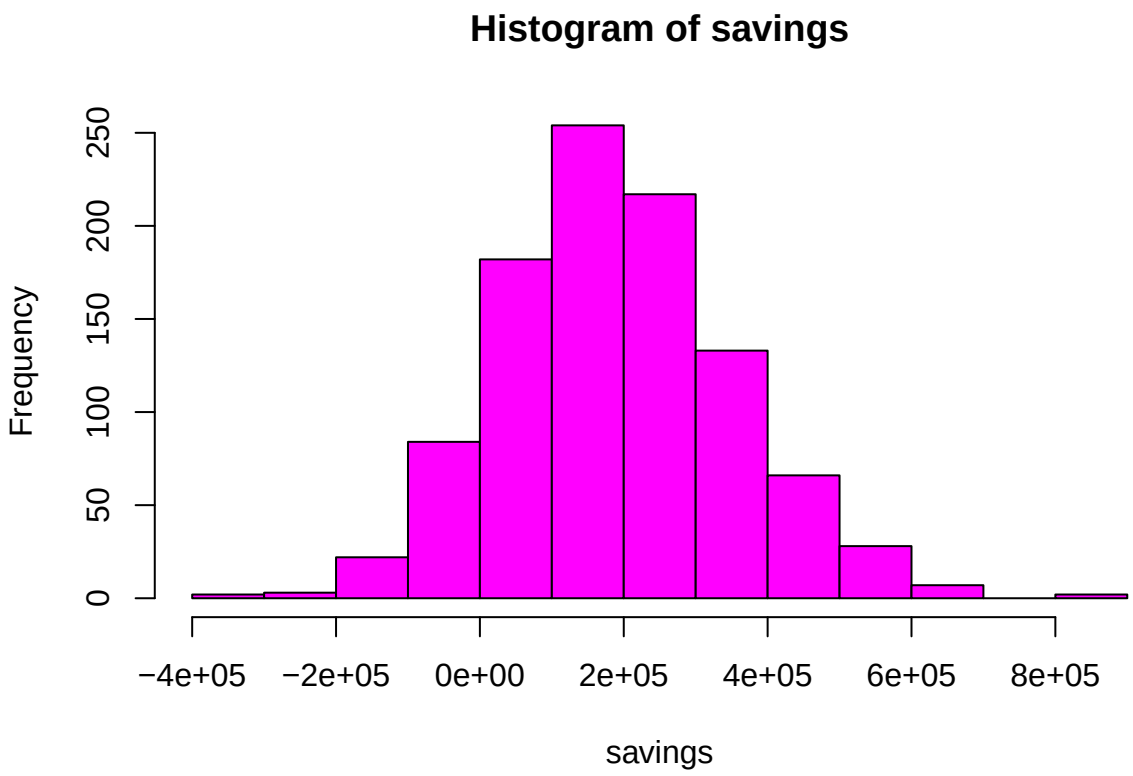
```
savings = rep(NA,1000)

for(i in 1:1000){
  units <- rnorm(1,40000,10000)
```

```
  save <- rnorm(units, 5,4)
  savings[i] = sum(save)
}

hist(savings,col="magenta")
```



Now if we don't use the means and get the product first...

```
savings = rep(NA,1000)

for(i in 1:1000){
  save <- rnorm(1,5,4)
  units <- rnorm(1,40000,10000)
  savings[i] = save*units
}

hist(savings,col="magenta")
```



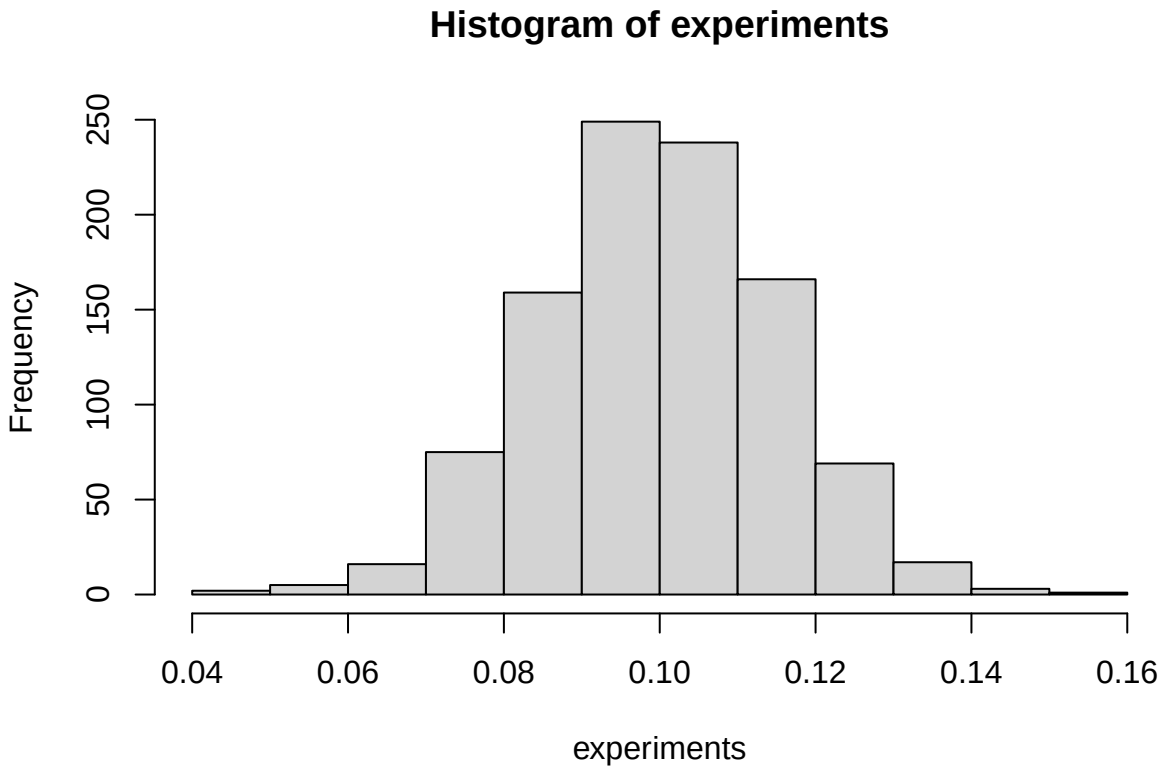
5.8 Coverage of confidence intervals:

On page 15 there is a discussion of an experimental study of an education-related intervention in Jamaica, in which the point estimate of the treatment effect, on the log scale, was 0.35 with a standard error of 0.17. Suppose the true effect is 0.10—this seems more realistic than the point estimate of 0.35—so that the treatment on average would increase earnings by 0.10 on the log scale. Use simulation to study the statistical properties of this experiment, assuming the standard error is 0.17.

(a) Simulate 1000 independent replications of the experiment assuming that the point estimate is normally distributed with mean 0.10 and standard deviation 0.17.

```
experiments = rep(NA,1000)
confint = rep(NA,1000)
confint0 = rep(NA,1000)

mean= 0.1
sd = 0.17

for (i in 1:1000) {
  sample = rnorm(127,0.1,sd)
  lower=mean(sample)+qt(0.025,126)*sd(sample)/sqrt(127)
  upper=mean(sample)+qt(0.975,126)*sd(sample)/sqrt(127)

  experiments[i] = mean(sample)
  confint[i] = ifelse(lower<0.1&upper>0.1,1,0)
  confint0[i] = ifelse(lower<0&upper>0,1,0)
```

```
}

hist(experiments)
```



(b) For each replication, compute the 95% confidence interval. Check how many of these intervals include the true parameter value.

```
sum(confint)
```

```
## [1] 955
```

(c) Compute the average and standard deviation of the 1000 point estimates; these represent the mean and standard deviation of the sampling distribution of the estimated treatment effect.

```
mean(experiments)
```

```
## [1] 0.09978979
```

```
sd(experiments)
```

```
## [1] 0.0151772
```

5.9 Coverage of confidence intervals after selection on statistical significance:

Take your 1000 simulations from Exercise 5.8, and select just the ones where the estimate is statistically significantly different from zero. Compute the average and standard deviation of the selected point estimates.

Compare these to the result from Exercise 5.8.

If they're significantly different from 0, the interval would not contain 0. In 5.8, I included a condition that checks if 0 is within our confidence interval. If 0 is in it, it's not significantly different.

```
significant <- experiments[confint0!=1]

mean(significant)
```

```
## [1] 0.09978979
```

```
sd(significant)
```

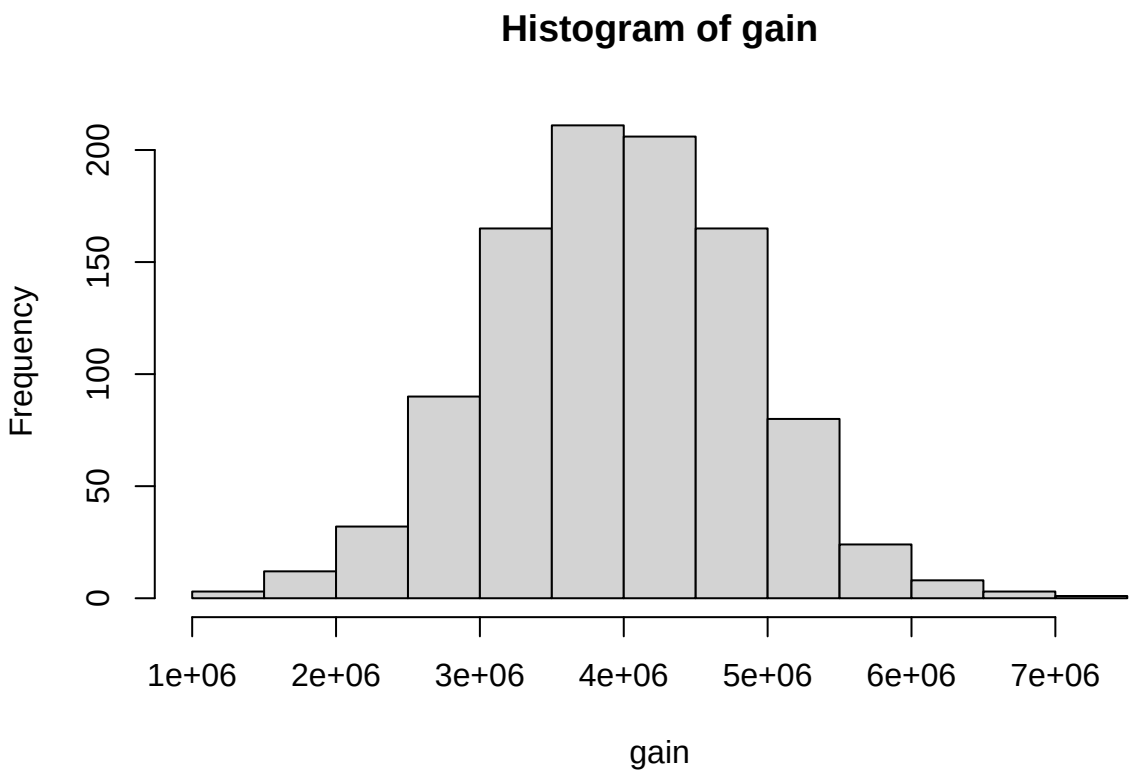
```
## [1] 0.0151772
```

None of our estimates seem

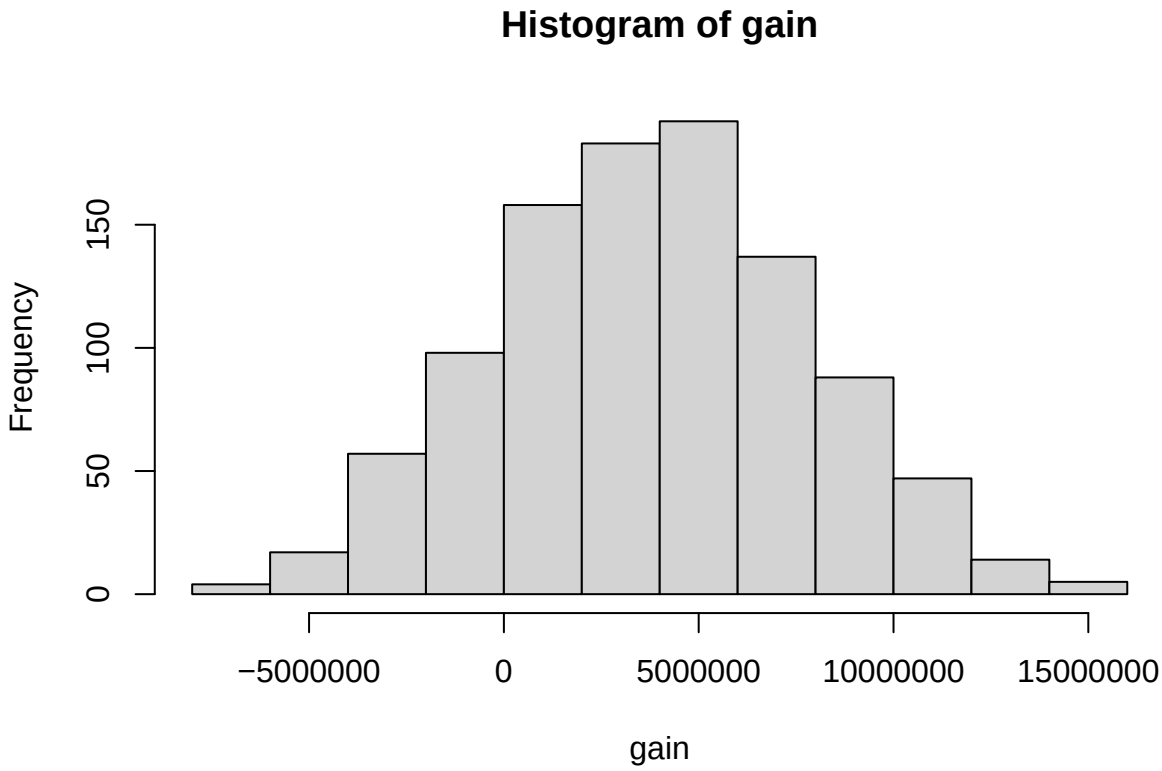
9.8 Simulation for decision analysis:

An experiment is performed to measure the efficacy of a television advertising program. The result is an estimate that each minute spent on a national advertising program will increase sales by 500,000 dollars, and this estimate has a standard error of 200000 dollars. Assume the uncertainty in the treatment effect can be approximated by a normal distribution. Suppose ads cost 300000 dollars per minute. What is the expected net gain for purchasing 20 minutes of ads? What is the probability that the net gain is negative?

```
cost = 300000*20

gain = rep(NA,1000)

for(i in 1:1000){
  value = rnorm(20,500000,200000)
  gain[i] = sum(value) - cost
}

for(i in 1:1000){
  value = rnorm(20,500000,200000)
  gain[i] = sum(value) - cost
}

hist(gain)
```




```
gain = rnorm(1000,500000,200000)*20 - 20*300000

hist(gain)
```



It seems unlikely from our histogram that 20 minutes of ads will have a negative net gain.

However, each ad has a 15% chance of having a negative net gain.

10.3 Checking statistical significance:

In this exercise and the next, you will simulate two variables that are statistically independent of each other to see what happens when we run a regression to predict one from the other. Generate 1000 data points from a normal distribution with mean 0 and standard deviation 1 by typing var1 <- rnorm(1000,0,1) in R. Generate another variable in the same way (call it var2). Run a regression of one variable on the other. Is the slope coefficient "statistically significant"? We do not recommend summarizing regressions in this way, but it can be useful to understand how this works, given that others will do so.

```
var1 <- rnorm(1000,0,1)
var2 <- rnorm(1000,0,1)
model = lm(var2~var1)
summary(model)
```

```
##
## Call:
## lm(formula = var2 ~ var1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3577 -0.6968  0.0127  0.6748  3.2138
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.01969    0.03237  -0.608    0.543
## var1         0.04019    0.03196   1.257    0.209
##
## Residual standard error: 1.023 on 998 degrees of freedom
## Multiple R-squared:  0.001582,   Adjusted R-squared:  0.0005811
## F-statistic: 1.581 on 1 and 998 DF,  p-value: 0.2089
```

In this sample, the slopes do not seem to be!

10.4 Simulation study of statistical significance:

Continuing the previous exercise, run a simulation repeating this process 100 times. This can be done using a loop. From each simulation, save the z-score (the estimated coefficient of var1 divided by its standard error). If the absolute value of the z-score exceeds 2, the estimate is "statistically significant." To perform this computation, we start by creating an empty vector of z-scores filled with missing values (NAs). Another approach is to start with z_scores <- numeric(length=100), which would set up a vector of zeroes. In general, however, we prefer to initialize with NAs, because then when there is a bug in the code, it sometimes shows up as NAs in the final results, alerting us to the problem.

How many of these 100 z-scores exceed 2 in absolute value, thus achieving the conventional level of statistical significance?

Here is code to perform the simulation:

This chunk will have eval=FALSE. If you want it to run, please copy the code to a new chunk, or remove eval=FALSE!

```
z_scores <- rep(NA,100)
for(k in 1:100) {
```

```
  var1 <- rnorm(1000,0,1)
  var2 <- rnorm(1000,0,1)
  fake <- data.frame(var1,var2)
  fit <- stan_glm(var2 ~ var1,data=fake,refresh=0)
  z_scores[k] <- coef(fit)[2] / se(fit)[2]
}
```

```
z_scores <- rep(NA,100)
for(k in 1:100) {
  var1 <- rnorm(1000,0,1)
  var2 <- rnorm(1000,0,1)
  fake <- data.frame(var1,var2)
  fit <- lm(var2 ~ var1,data=fake)
  z_scores[k] <- summary(fit)$coefficients[2,1]/summary(fit)$coefficients[2,2]
}

sum(z_scores>2)
```

```
## [1] 4
```

11.3 Coverage of confidence intervals:

Consider the following procedure:

- Set n = 100 and draw n continuous values x_i uniformly distributed between 0 and 10. Then simulate data from the model $y_i = a + bx_i + error_i$, for $i = 1, \ldots, n$, with $a = 2$, $b = 3$, and independent errors from a normal distribution.
- Regress y on x. Look at the median and mad sd of b. Check to see if the interval formed by the $median \pm 2$ mad sd includes the true value, $b = 3$.
- Repeat the above two steps 1000 times.

(a) True or false: the interval should contain the true value approximately 950 times. Explain your answer.

True. We use lm instead of stan_lm for computational convinence but the result should be fairly comparable.

```
conf <- c(1:1000)

for(i in 1:1000) {
n = 100
x <- runif(n,0,10)
y = 2 + 3*x + rnorm(n,0,3)
fit = lm(y~x)

lower= summary(fit)$coefficients[2,1] - 2*summary(fit)$coefficients[2,2]
upper = summary(fit)$coefficients[2,1] + 2*summary(fit)$coefficients[2,2]
conf[i] = ifelse(lower<3&upper>3,1,0)
}

mean(conf)
```

```
## [1] 0.944
```

(b) Same as above, except the error distribution is bimodal, not normal. True or false: the interval should contain the true value approximately 950 times. Explain your answer.

The assumption is not meat, so it's possible that this might not work. However, the answer turns out to be true! Why? Let's look at the simulation.

```
conf <- c(1:1000)

for(i in 1:1000) {
n = 100
x <- runif(n,0,10)
w <- rbinom(n,1,0.2)
mn<-ifelse(w==1,-11, 10 )
y = 2 + 3*x + rnorm(n,mn,2)
fit2 = lm(y~x)

lower= summary(fit2)$coefficients[2,1] - 2*summary(fit2)$coefficients[2,2]
upper = summary(fit2)$coefficients[2,1] + 2*summary(fit2)$coefficients[2,2]
conf[i] = ifelse(lower<3&upper>3,1,0)
}

mean(conf)
```

```
## [1] 0.956
```

Higher variance in answers but a lot of the time, still ~950 of them seem to pass.

The marginal model plot shows why this is the case. Since we assume the errors are independently bimodal, the fitted line is shifted but it still gets the slope correctly.

```
library(car)
```

```
## Loading required package: carData
```

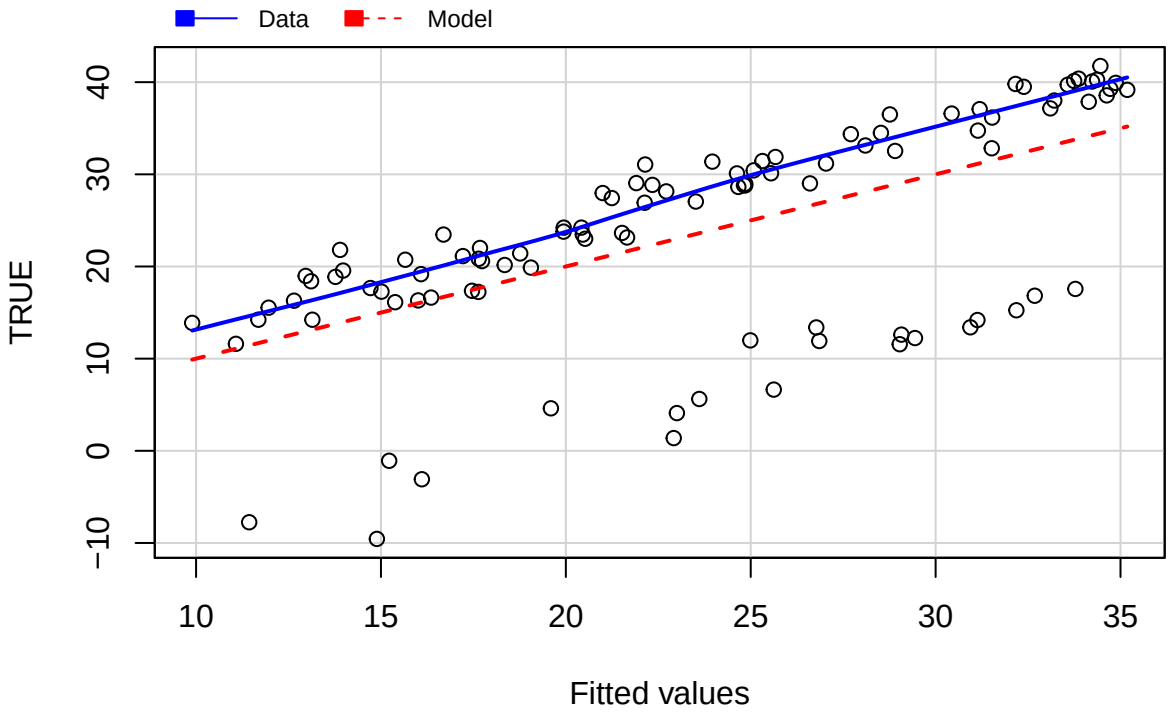
```
## Registered S3 methods overwritten by 'car':
##   method                           from
##   influence.merMod                 lme4
##   cooks.distance.influence.merMod lme4
##   dfbeta.influence.merMod          lme4
##   dfbetas.influence.merMod         lme4
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:rstanarm':
##
##     logit
```

```
## The following object is masked from 'package:arm':
##
##     logit
```

```
marginalModelPlot(fit2)
```



There are ways to break this by assuming the level of bimodality depends on the value of x. But we will leave it for now.

Optional:

11.6 Fitting a wrong model:

Suppose you have 100 data points that arose from the following model: $y = 3 + 0.1x1 + 0.5x2 + error$, with independent errors drawn from a t distribution with mean 0, scale 5, and 4 degrees of freedom. We shall explore the implications of fitting a standard linear regression to these data.

(a) Simulate data from this model. For simplicity, suppose the values of x1 are simply the integers from 1 to 100, and that the values of x2 are random and equally likely to be 0 or 1. In R, you can define `x_1 <- 1:100`, simulate `x_2` using `rbinom`, then create the linear predictor, and finally simulate the random errors in y using the rt function. Fit a linear regression (with normal errors) to these data and see if the 68% confidence intervals for the regression coefficients (for each, the estimates ±1 standard error) cover the true values.

```
x1 <- 1:100
x2 = rbinom(100,1,0.5)
error = rt(100,df=4)

y = 3 + .1*x1 + .5*x2 + error

model = lm(y~x1+x2)
model
```

```
##
## Call:
## lm(formula = y ~ x1 + x2)
```

```
##
## Coefficients:
## (Intercept)           x1           x2
##     2.83819      0.09759      0.87322
```

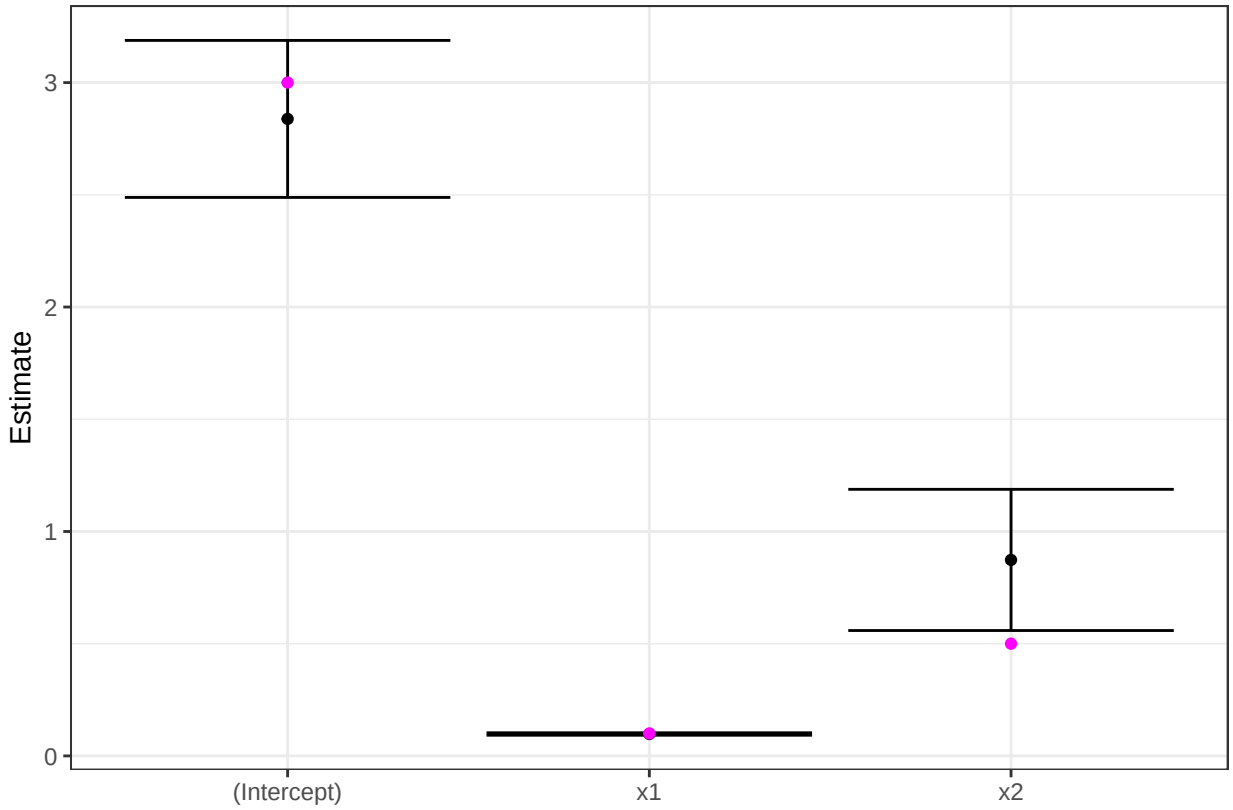
```
coefs = data.frame(summary(model)$coefficients)
coefs
```

```
##               Estimate  Std..Error   t.value     Pr...t..
## (Intercept) 2.83818783 0.349785836  8.114073 1.534907e-12
## x1          0.09759419 0.005421675 18.000745 1.102559e-32
## x2          0.87322258 0.314581773  2.775821 6.607900e-03
```

```
coefs$Estimate
```

```
## [1] 2.83818783 0.09759419 0.87322258
```

```
ggplot(coefs) + geom_point(aes(x=rownames(coefs),y=Estimate)) +
  geom_errorbar(aes(x=rownames(coefs),
                    ymin=Estimate-Std..Error,ymax=Estimate+Std..Error)) + labs(x="") +
  geom_point(aes(x=rownames(coefs),y=c(3,.1,.5)),color="magenta") + theme_bw()
```



We can see that we capture x1 and our intercept, but not x2!

(b) Put the above step in a loop and repeat 1000 times. Calculate the confidence coverage for the 68% intervals for each of the three coefficients in the model.

```
x1=1:100
intconf = data.frame(mean=rep(NA,1000),upper=rep(NA,1000),lower=rep(NA,1000))
```

```
x1conf = data.frame(mean=rep(NA,1000),upper=rep(NA,1000),lower=rep(NA,1000))
x2conf = data.frame(mean=rep(NA,1000),upper=rep(NA,1000),lower=rep(NA,1000))

for(i in 1:1000){
  x2= rbinom(100,1,0.5)
  error=rt(100,df=4)
  y= 3 + .1*x1 + .5*x2 + error
  model = lm(y~x1+x2)
  coefs = summary(model)$coefficients

  intconf$mean[i] = coefs[1,1]
  x1conf$mean[i] = coefs[2,1]
  x2conf$mean[i] = coefs[3,1]

  intconf$upper[i] = coefs[1,1] + coefs[1,2]
  intconf$lower[i] = coefs[1,1] - coefs[1,2]

  x1conf$upper[i] = coefs[2,1] + coefs[2,2]
  x1conf$lower[i] = coefs[2,1] - coefs[2,2]

  x2conf$upper[i] = coefs[3,1] + coefs[3,2]
  x2conf$lower[i] = coefs[3,1] - coefs[3,2]
}

mean(intconf$upper>3&intconf$lower<3)
```

```
## [1] 0.684
```

```
mean(x1conf$upper>0.1&x1conf$lower<0.1)
```

```
## [1] 0.67
```

```
mean(x2conf$upper>0.5&x2conf$lower<0.5)
```

```
## [1] 0.664
```

11.9 Leave-one-out cross validation:

Use LOO to compare different models fit to the beauty and teaching evaluations example from Exercise 10.6:

```
beauty <- read.csv("https://raw.githubusercontent.com/avehtari/ROS-Examples/master/Beauty/data/beauty.cs
                   header=T)
M10.6 <- stan_glm(eval ~ beauty,data=beauty,refresh=0)
M10.6b <- stan_glm(eval ~ beauty + female + beauty:female,data=beauty,refresh=0)

loo1 = loo(M10.6)
loo2 = loo(M10.6b)
```

(a) Discuss the LOO results for the different models and what this implies, or should imply, for model choice in this example.

```
loo1
```

```
##
```

```
## Computed from 4000 by 463 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -378.3 14.6
## p_loo         2.9  0.3
## looic       756.5 29.3
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

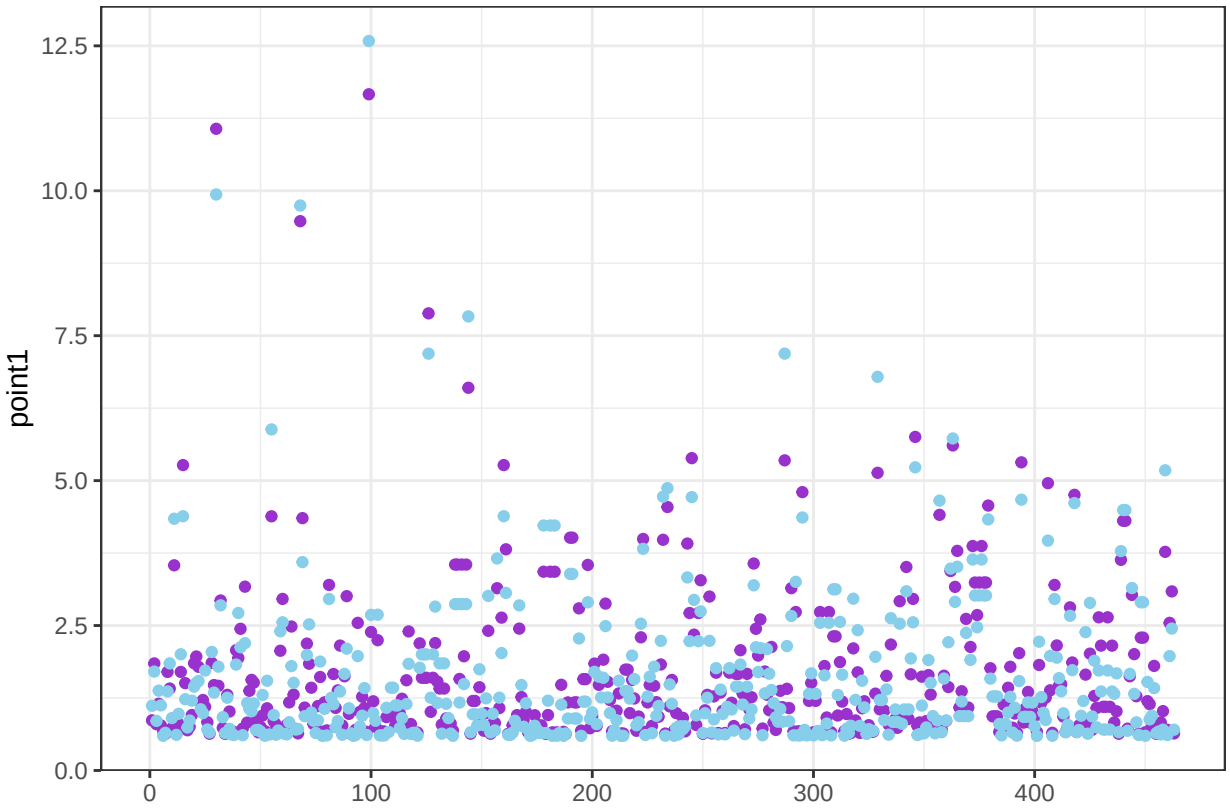
```
loo2
```

```
##
## Computed from 4000 by 463 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -371.3 14.9
## p_loo         4.9  0.4
## looic       742.6 29.7
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

Our output doesn't change by much, but seems to suggest we pick the second model, due to the lower looic.

(b) Compare predictive errors pointwise. Are there some data points that have high predictive errors for all the fitted models?

```
errdisplay = data.frame(x=1:nrow(beauty),point1=loo1$pointwise[,4],point2=loo2$pointwise[,4])

ggplot(errdisplay,aes(x=x)) + geom_point(aes(y=point1),color="darkorchid") +
  geom_point(aes(y=point2),color="skyblue") + theme_bw() + labs(x="")
```



There seem to be some points doing worse for both!

11.10 K-fold cross validation:

Repeat part (a) of the previous example, but using 5-fold cross validation:

(a) Randomly partition the data into five parts using the sample function in R.

```
beauty$group = sample(1:5,nrow(beauty),replace=T)
```

(b) For each part, re-fitting the model excluding that part, then use each fitted model to predict the outcomes for the left-out part, and compute the sum of squared errors for the prediction.

```
model = lm(eval~beauty,data=beauty[beauty$group!=i,])

SSE1 = rep(NA,5)
for(i in 1:5){
  model = lm(eval~beauty,data=beauty[beauty$group!=i,])
  SSE1[i] = sum((beauty[beauty$group==i,]$eval - predict(model,newdata=beauty[beauty$group==i,]))^2)
}

SSE2 = rep(NA,5)
for(i in 1:5){
  model = lm(eval~beauty + female + beauty:female,data=beauty[beauty$group!=i,])
```

```
  SSE2[i] = sum((beauty[beauty$group==i,]$eval - predict(model,newdata=beauty[beauty$group==i,]))^2)
}
```

(c) For each model, add up the sum of squared errors for the five steps in (b). Compare the different models based on this fit.

```
sum(SSE1)-sum(SSE2)
```

```
## [1] 3.567943
```

Our first model seems to have a slightly higher SSE, so we should pick model 2!